The Maximum Growth of Money in Vanilla Balatro

July 27, 2026

Abstract

If you don't want to verify this along with me, here are the actual answers. First, I present the solved theoretical growth function;

$$\text{Money}_r \approx (7.49 \times 10^{13}) \cdot \left(\frac{\text{Money}_0}{7.49 \times 10^{13}} \right)^{2^r}$$

This grows at a Big O speed of 2^{2^x}, much faster than any other scaling in the game, not including the scaling (such as scoring) that depends on this growth. This doesn't tell you how your money will change from round to round, however. To find the arbitrary growth, values that shrink to zero as r grows get removed. To find exactly how your money should change per round, we need the recurrence relation. That looks like this:

$$\text{Money}_{r+1} = \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \left(\sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r} + \frac{3}{700} \left(\sqrt{8 \cdot \text{Money}_r + 1} + 1 \right) \right)^4$$

Alright, where are these numbers coming from?

Restrictions

This run is not a theoretically maximum run, just the maximum if we're talking about growth. We have two major restrictions, and I'll justify them here, and talk about alternatives to these restrictions.

0.1 Perkeo is Banned

Perkeo has a very well known and exploitable infinite. Get every single tarot in negative form minus the Fool, Emperor, and one of [Judgement|Hermit|Temperance]. Use the Emperor to spawn a Fool and one of those, use the Fool to spawn an Emperor, and repeat. This has no limits. Growth breaks down in meaning, as the only growth limit is how much time you're willing to spend.

0.1.1 Alternative 1: Pseudo-Tarot Looping

This is one alternative if you're trying for "no infinites." Ban the Fool from copying the Emperor. If the Fool can't copy the Emperor, there is no infinite. There have been runs using this limitation, the only one I know of is by @DrDreaMO. This paper will not explore this possibility.

0.1.2 Alternative 2: Redefine Progression by Time Instead of Rounds

We could try and find how fast your money grows by time spent instead of by round. Infinites can roll the shop forever, but is that the fastest way to grow your money? I'm not going to explore this question.

0.2 Profiles are Fully Unlocked

Again, we could strategically leave some jokers locked. The rare joker pool could be cut in half, the uncommon pool can be significantly trimmed, and a few common jokers can be left out as well. We could also play on a set seed, stopping these jokers from being unlocked naturally as the conditions are met as we progress through the run. We're not doing that.

0.2.1 Alternative 1: Seeded & Strategically Managed Profiles

This is technically optimal. Locked jokers will never be unlocked. We'll get way more of every useful joker. The downside is, it makes this run completely impossible to achieve under normal conditions. You can never start up a normal run and achieve these conditions, you need to play seeded.

0.2.2 Alternative 2: Unseeded & Strategically Managed Profiles

This is better and worse. This makes the run theoretically viable in normal gameplay, as not every player will have everything unlocked. You could just happen to have a very well managed profile by pure chance, start up a run, and achieve these limits. The downside is this makes growth hard to generalize, as we don't know which jokers you have unlocked, and we don't know when your run will unlock jokers. We will not be doing this either.

1 Finding the Theoretical Limits of Money Generation

In order to figure this out, we need to know how money grows as you advance through the game. You get money through jokers by advancing through shops. This is capped quite early by your joker slots in practice. In theory, however, those jokers can be negative, bypassing joker slots entirely. So in order to find out how money theoretically grows, we need to know how money turns into shop slots, how shop slots turn into negative jokers, and how negative jokers turn back into money.

1.1 Money into Shop Slots

Money can be converted into rerolls in the following increasing manner; \$1 gives you a reroll, \$2 gives you another, \$3 gives you another, and so on. Anyone familiar with calculating triangle numbers will remember this relationship.

$$\text{Slots} = 4 \cdot \text{Rerolls} \iff \text{Rerolls} = \frac{\text{Slots}}{4}$$

$$\begin{aligned}
\text{Money}_r &= \sum_{i=1}^{\text{Rerolls}} \text{x} = \$1 + \$2 + \$3 + \cdots + \$\text{final} \\
&= \frac{\text{Rerolls}(\text{Rerolls}+1)}{2} \\
&= \frac{\frac{\text{Slots}}{4}(\frac{\text{Slots}}{4}+1)}{2} \\
&= \frac{\text{Slots}(\text{Slots}+4)}{32}
\end{aligned}$$

We now have our Money in terms of how many Shop Slots we'll see from spending it. Reversing it tells us how many Shop Slots we'll see from a given amount of money.

$$\begin{aligned}
\text{Money} &= \frac{\text{Slots}(\text{Slots}+4)}{32} \\
32 \cdot \text{Money} + 4 &= \text{Slots}^2 + 4 \cdot \text{Slots} + 4 + 4 \\
32 \cdot \text{Money} + 4 &= (\text{Slots}+2)^2 \\
\sqrt{32 \cdot \text{Money} + 4} &= \text{Slots} + 2 \\
\text{Slots} &= 2\left(\sqrt{8 \cdot \text{Money} + 1} - 1\right) + \underbrace{4}_{\text{Initial Shop State}}
\end{aligned}$$

1.2 Shop Slots into Useful Negative Jokers

This one is extremely simple. It's a flat relationship, nothing special going on. It is different for every rarity of joker, and we will be using every rarity of joker, so we'll go ahead and find them all. For some context, every slot has a $\frac{20}{28}$ probability of producing a joker, and each of those jokers

has a $\frac{3}{1000}$ chance of being negative. Then we find the rarity of the produced joker, then the size of the pool of jokers within that rarity.

$$\text{Specific Joker} = \text{Slots} \cdot \text{Chance of Joker} \cdot \text{Chance of Negative} \cdot \text{Chance of Rarity} \cdot \frac{1}{\text{Size of Pool}}$$

$$\begin{aligned}
\text{Common} &= \text{Slots} \cdot \frac{20}{28} \cdot \frac{3}{1000} \cdot \frac{7}{10} \cdot \frac{1}{60} = \frac{1}{40{,}000}\text{Slots} \\
\text{Uncommon} &= \text{Slots} \cdot \frac{20}{28} \cdot \frac{3}{1000} \cdot \frac{1}{4} \cdot \frac{1}{60} = \frac{1}{112{,}000}\text{Slots} \\
\text{Rare} &= \text{Slots} \cdot \frac{20}{28} \cdot \frac{3}{1000} \cdot \frac{1}{20} \cdot \frac{1}{20} = \frac{3}{560{,}000}\text{Slots}
\end{aligned}$$

1.3 Useful Negatives Back into Money

Reserved Parking is the fastest money scaler in the game. It's the only thing that operates at a level of x^4, as it scales based on your Reserved Parking triggers, your retriggers, your total hands available[1], *and* your hand size. All of these can be pushed forwards by negative jokers.

This sounds simple, but we have a problem. How do we account for the increased hand size from DNAs and Certifications?

1.3.1 DNA and Certifications

In order to do this, we're going to shift from thinking of hands and hand size as different things, and instead think of them in the terms of total cards operated on. Cards added temporarily to the hand at the beginning of the round would start at the total number of cards generated, then slowly go to zero as hands get played. This can then be added to our standing total of cards operated on, which is simply Hands · Hand Size.

$$\text{Cards In Hand from Generators} = \sum_{i=1}^{\text{Total Cards Created (C)}} x = 1 + 2 + 3 + \cdots + C \approx \frac{C^2}{2}$$

$$\text{Total Cards Across All Hands} = \text{Hands} \cdot \text{Hand Size} + \frac{C^2}{2}$$

Now that we have our money generation in terms we can work with, which is our total cards in hand across all hands, reserved parking triggers, and retriggers, it's time to start thinking of money in terms of jokers. Once we can do that, we can start thinking of money in terms of shop slots, and then back into money.

$$\begin{aligned}
\text{Money} &= \text{Total Cards Across All Hands} \cdot \text{Reserved Parking Triggers} \cdot \text{Retriggers} \\
C &= (\text{Certificate} + \text{Copy}) + (\text{DNA} + \text{Copy}) \\
&= \frac{1}{28{,}000} \cdot \text{Slots} \\
\text{Money} &= \left(\underbrace{3 \cdot (\text{Copy} + \text{Burglar})}_{\text{Hands}} \cdot \underbrace{(2 \cdot \text{Troubadour} + \text{Juggler})}_{\text{Hand Size}} + \frac{C^2}{2} \right) \cdot \text{Reserved} \cdot \underbrace{(\text{Copy} + \text{Mime})}_{\text{Retriggers}} \\
\text{Money} &= \left(\frac{33}{560{,}000} \cdot \text{Slots} \cdot \frac{3}{70{,}000} \cdot \text{Slots} + \frac{1}{784{,}000{,}000} \cdot \text{Slots}^2 \right) \cdot \left(\frac{1}{40{,}000} \cdot \text{Slots} \right) \cdot \left(\frac{11}{560{,}000} \cdot \text{Slots} \right) \\
\text{Money} &= \frac{143}{10{,}976{,}000{,}000{,}000{,}000{,}000} \cdot \text{Slots}^4
\end{aligned}$$

[1] The astute will notice that we ignore Troubadours here. The hand reduction of Troubadours happens strictly before Burglar triggers, and can't lower your hands below 1. This means in any vanilla context, any Troubadours beyond 6 will never impact the amount of hands you can play

This would tell us how much money we'll make next round, but we need to remember that we already have some jokers that are producing money for us currently. This won't be reflected in the growth function, as when 2^{2^x} gets going what happened the previous round quickly becomes irrelevant, but for the recurrence relation it's vital. To add those in, we need to extract out our total joker pool from this function.

$$\begin{aligned}
\text{Total Negative Jokers(J)} &= \frac{20}{28} \cdot \frac{3}{1{,}000} \cdot \text{Slots} \\
\text{Slots} &= \text{J} \cdot \frac{1{,}400}{3} \\
\text{Money} &= \frac{143}{10{,}976{,}000{,}000{,}000{,}000{,}000} \cdot \text{Slots}^4 \\
&= \frac{143}{10{,}976{,}000{,}000{,}000{,}000{,}000} \cdot \left(\text{J} \cdot \frac{1{,}400}{3}\right)^4 \\
&= \frac{143}{10{,}976{,}000{,}000{,}000{,}000{,}000} \cdot \left(\frac{3{,}841{,}600{,}000{,}000}{81}\right) \cdot \text{J}^4 \\
&= \frac{549{,}348{,}800{,}000{,}000}{889{,}056{,}000{,}000{,}000{,}000{,}000} \cdot \text{J}^4 \\
&= \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \text{J}^4
\end{aligned}$$

This final equation shows us how much money we can expect to make from any given pool of negative jokers, assuming the ratios of the negative jokers follow the ratios that you should expect to see them.

1.4 Finding Our Total Jokers

What's great about this is that reversing it gives us the joker pool we used to get the money we're spending in any given round. Combined with our previous work, we've successfully found out what our joker pool will look like in any given round.

$$\begin{aligned}
\text{J}_\text{r} &= \sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r} \\
\text{Slots}_r &= 2\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right) \\
\text{J}_\text{new} &= \frac{3}{1{,}400} \cdot \text{Slots}_r \\
&= \frac{3}{1{,}400} \cdot 2 \cdot \left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right) \\
\text{J}_{r+1} &= \text{J}_\text{r} + \text{J}_\text{new} \\
&= \underbrace{\sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r}}_{\text{r}} + \underbrace{\frac{3}{700} \cdot \left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right)}_{\text{new}}
\end{aligned}$$

1.5 The Final Recurrence Relation

This is just plugging in the final pieces. We've already seen how to convert a pool of jokers into money. Now we need to do that for the joker pool we've calculated we can expect in the next round.

$$\text{Money}_{r+1} = \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \text{J}_{r+1}^4$$

$$\boxed{\text{Money}_{r+1} = \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \left(\sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r} + \frac{3}{700}\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right)\right)^4}$$

1.6 Solving The Recurrence Relation

First, we need to pare away everything that dies when these numbers get extremely large. The old jokers become completely irrelevant, as do any static additions to the numbers. Let's take a look at how that looks.

$$\begin{aligned}
\text{Money}_{r+1} &= \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \left(\underbrace{\sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r}}_{\text{Becomes Irrelevant}} + \frac{3}{700}\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right) \right)^4 \\
&\approx \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \left(\frac{3}{700}\left(\sqrt{8 \cdot \text{Money}_r}\right) \right)^4 \\
&\approx \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \frac{3^4 \cdot \left(\sqrt{8}^4\right)}{700^4} \cdot \text{Money}_r^2 \\
&\approx X \cdot \text{Money}_r^2 \\
\text{where } X &= 1.33 \times 10^{-14}
\end{aligned}$$

This simplified version now needs to get solved, meaning for any initial value we should be able to approximate not just the next round, but any arbitrary number of rounds. To do that, first we should clean this up a bit.

$$\begin{aligned}
\text{Money}_{r+1} &= X \cdot \text{Money}_r^2 \\
\log_2(\text{Money}_{r+1}) &= \log_2(X) + 2\log_2(\text{Money}_r) \\
Y_r &= \log_2(\text{Money}_r) \\
C &= \log_2(X) \\
Y_{r+1} &= 2Y_r + C
\end{aligned}$$

This recurrence relation is much easier to solve. Let's take a look at how it behaves as we move through the steps, so we can turn it into a generalized form.

$$\begin{aligned}
&\text{Round 1 } (r=1): && Y_1 = 2Y_0 + C && \\
&\text{Round 2 } (r=2): && Y_2 = 2Y_1 + C && = 2(2Y_0 + C) + C && = 4Y_0 + 3C \\
&\text{Round 3 } (r=3): && Y_3 = 2Y_2 + C && = 2(4Y_0 + 3C) + C && = 8Y_0 + 7C \\
&\text{Round r}: && Y_r = 2^r Y_0 + (2^r - 1)C &&
\end{aligned}$$

Perfect. Let's substitute back in our money, and finally solve this thing.

$$\begin{aligned}
\log_2(\text{Money}_r) &= 2^r \log_2(\text{Money}_0) + (2^r - 1)\log_2(X) \\
\text{Money}_r &= (\text{Money}_0)^{2^r} \cdot X^{(2^r - 1)} \\
&= \frac{1}{X}\left(X \cdot \text{Money}_0\right)^{2^r} \\
&\boxed{\text{Money}_r \approx (7.49 \times 10^{13}) \cdot \left(\frac{\text{Money}_0}{7.49 \times 10^{13}}\right)^{2^r}}
\end{aligned}$$

2 The Problem of the Deck

We've ignored something for too long. The above doesn't take your deck size into account at all, because it's much harder to account for purely using the money you've gained the previous round. We've been hand-waving it away, expecting you to have an infinite deck. That stops now. Let's look at how we can expect our deck to change as we grow our joker pool.

2.1 Joker Ratios

First, let's look at how our hands, our hand size, and our deck grow with relation to our negative jokers. We need to look at this through the lens of ratios, so how common the joker in question is and how big the pool of that rarity is.

$$\begin{aligned}
\text{Hand Size Jokers} &= 2 \cdot \text{Troubadours} + \text{Jugglers} \\
&= 2 \cdot \left(\frac{1}{4} \cdot \frac{1}{60}\right) + \left(\frac{7}{10} \cdot \frac{1}{60}\right) \\
&= \frac{24}{1{,}200} \\
\text{Hand Jokers} &= 3 \cdot (\text{Burglars} + \text{Copies}) \\
&= 3 \cdot \left(\frac{1}{4} \cdot \frac{1}{60} + \frac{1}{20} \cdot \frac{2}{20}\right) \\
&= \frac{33}{1{,}200} \\
\text{Card Generation Jokers} &= \text{Marbles} + (\text{Certificate} + \text{Copies}) + (\text{DNAs} + \text{Copies}) \\
&= \frac{1}{4} \cdot \frac{1}{60} + \left(\frac{1}{4} \cdot \frac{1}{60} + \frac{1}{20} \cdot \frac{2}{20}\right) + \left(\frac{1}{20} \cdot \frac{1}{20} + \frac{1}{20} \cdot \frac{2}{20}\right) \\
&= \frac{25}{1{,}200}
\end{aligned}$$

33 Hands | 25 Cards | 24 Hand Size

You'll notice two things here. First, our primary method of card generation, Certificates and DNAs, don't care about your hand size. Second, the amount of additional hands we can play every round outpaces the amount of additional cards you can generate.

This means neither the hands we can play nor our hand size functionally matter. They are both capped not by Burglars and Troubadours, but by our card generation. Once this engine begins only the cards that we make during the current round functionally matter, so every round will begin by us drawing our entire deck to our hand and then playing our hands over an an ever-diminishing hand size.

2.2 Revisiting our Equation for Money Generation

$$\text{Money}_{old} = 3 \cdot (\text{Copy} + \text{Burglar}) \cdot \text{Reserved} \cdot (\text{Copy} + \text{Mime}) \cdot (2 \cdot \text{Troubadour} + \text{Juggler})$$

This was what we previously calculated. You'll see that our hands are incorrectly capped by our Burglars and our Copies, and our hand size is incorrectly capped by our Troubadours and Jugglers. Let's fix that.

$$\text{Money}_{fixed} = \frac{1}{2} \cdot (\text{Marble} + \text{Certificate} + \text{DNA} + 2 \cdot \text{Copy})^2 \cdot \text{Reserved} \cdot (\text{Copy} + \text{Mime})$$

Both our hands and our hand size are capped by the amount of cards we can generate in a single hand. This is halved, which we can see if you look at the exact middle hand we play. Our entire deck is in our hands, and we've played half the cards, so Reserved Parking and Mime only operate on half the cards we generated.

If you look at the hands we played before and after this middle hand, the -1 and +1 cards we'll have in hand cancel each other out, and it's functionally the same as playing 3 of that middle hand. We can stretch this out to cover all of our hands, and see that playing a single card at a time, slowly diminishing your hand size until it hits zero, is the same as playing all of those hands at half the strength you had when your hand was full.

2.3 Correcting for this New Equation

$$\begin{aligned}
\text{Money} &= \frac{1}{2} \cdot (\text{Marble} + \text{Certificate} + \text{DNA} + 2 \cdot \text{Copy})^2 \cdot \text{Reserved} \cdot (\text{Copy} + \text{Mime}) \\
&= \frac{1}{2} \cdot \left(\frac{1}{22{,}400} \cdot \text{Slots}\right)^2 \cdot \left(\frac{1}{40{,}000} \cdot \text{Slots}\right) \cdot \left(\frac{11}{560{,}000} \cdot \text{Slots}\right) \\
&= \frac{11}{22{,}478{,}848{,}000{,}000{,}000{,}000} \cdot \text{Slots}^4 \\
&= \frac{11}{22{,}478{,}848{,}000{,}000{,}000{,}000} \cdot \left(\frac{3{,}841{,}600{,}000{,}000}{81}\right) \cdot \text{J}^4 \\
&= \frac{77}{3{,}317{,}760{,}000} \cdot \text{J}^4
\end{aligned}$$

$$\text{Money}_{r+1} = \frac{77}{3{,}317{,}760{,}000} \cdot \left(\sqrt[4]{\frac{3{,}317{,}760{,}000}{77} \cdot \text{Money}_r} + \frac{3}{700}\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right)\right)^4$$

$$X = \frac{77}{3{,}317{,}760{,}000} \cdot \frac{3^4 \cdot \left(\sqrt{8}^4\right)}{700^4} \approx 5.01 \times 10^{-16}$$

$$\boxed{\text{Money}_r \approx (1.99 \times 10^{15}) \cdot \left(\frac{\text{Money}_0}{1.99 \times 10^{15}}\right)^{2^r}}$$

3 Double Tags and Negative Skips: Better at Lower Values

The above details how money grows in Balatro **using a Showman.** This is because a Showman is the only way to endlessly grow your money within the bounds of a single shop. However, there is a secondary way to generate value. Negative skips.

Diet Colas are extremely powerful. They give you a double tag. This gives a guaranteed negative so long as you find a negative skip, bypassing the 0.3% chance to find one naturally. As they can be sold, they can be used alongside Joker blocking. This gives them a lot of advantages for the beginning of a run.

Unfortunately, cashing in those double tags requires you skip a round. As the 2^{2^r} growth kicks in, this becomes less and less attractive, no matter how much more efficient they are at gathering negative jokers. However, this efficiency allows them to be extremely useful for times when your initial money generated isn't large enough to effectively utilize our growth function.

In order to know when to use this new growth, we first need to find it. How does maximizing for Diet Colas grow your money?

3.1 Chaos the Clown

A Showman is no longer assumed. Through the accumulation of negative jokers, we can shrink the pools of jokers found in rarities to whatever we like. In the Common rarity, there is a clear winner; Chaos the Clown. A free reroll that doesn't increase the cost of future rerolls is extremely powerful. How do we maximize this?

The unfortunate truth is, this is more complicated than it appears at first glance. The temporary blocking functions of both the Chaos the Clown and the Judgements we'll be holding in our consumable slots make this non-trivial to answer.

To find this answer, we need to build a state system.

3.1.1 Isolating the States

What States can the shop be in? Four states can be isolated.

1. Default(D) = Chaos the Clown, Judgements and Fools can Spawn.

2. Free No Inventory(F_0) = Chaos the Clown can't spawn, Judgements and Fools can Spawn.
3. Free 1 Inventory(F_1) = Chaos the Clown and Judgements can't spawn, Fools can Spawn, 1 Judgement is held.
4. Free 2 Inventory(F_2) = Chaos the Clown and Judgements can't spawn, Fools can Spawn, 2 Judgements are held.

Something is very apparent looking at these states. Only one is paid for, the other three are free. All of them represent a valid condition for a given reroll to end up in. Therefore, let's look at this in terms of traffic.

We can find the traffic from every state to every other state. We are only concerned with what happens per paid reroll, so we can set the traffic of the Default state to 1. Then, using that, we can find the traffic we'll expect to see from that 1 default state. This should allow us to see how many total states we'll be visiting on every paid reroll.

3.1.2 Finding the Relevant Shop Slot Probabilities

From here, we need to find how these states behave. To begin, we need to find the probabilities a Shop Slot contains a Judgement, Fool, Chaos, or nothing.

Blocking is the act of holding an item simply so it no longer spawns in the shop. We will be doing this with a negative version of every Common joker besides Chaos the Clown, and a single non-negative Tarot we'll hold in our third consumable slot. Now we only have a single Common joker in the pool, and 21 available Tarots.

The only edge case we need to consider is what happens when we enter a shop holding a Judgement. Now, the Fool has 2 Tarots limiting the pool, so it is slightly more likely to spawn. Similarly, the chance of a miss changes depending on whether a Chaos can spawn in a shop, and whether a judgement can spawn in a shop. There are no situations we'll find ourselves in where a Judgement cannot spawn but a Chaos can, which simplifies things somewhat.

$$
\begin{aligned}
S_{\text{Chaos(C)}} &= \frac{20}{28} \cdot \frac{7}{10} = \frac{1}{2} \\
S_{\text{Judgement(J)}} &= \frac{4}{28} \cdot \frac{1}{21} = \frac{1}{147} \\
S_{\text{Fool(F)}} &= \frac{4}{28} \cdot \frac{1}{21} = \frac{1}{147} \\
S_{\text{Fool without Judgement(Fj)}} &= \frac{4}{28} \cdot \frac{1}{20} = \frac{1}{140} \\
S_{\text{Miss(M)}} &= 1 - S_{\text{C}} - S_{\text{J}} - S_{\text{F}} = \frac{143}{294} \\
S_{\text{Miss without Chaos(Mc)}} &= 1 - S_{\text{J}} - S_{\text{F}} = \frac{145}{147} \\
S_{\text{Miss without Judgements(Mj)}} &= 1 - S_{\text{Fj}} = \frac{139}{140}
\end{aligned}
$$

3.1.3 Finding the Relevant Full Shop Probabilities

We now need to find out how likely it is to reach any of these states from any of the other states.

First, we need to know how likely it is to find any combination of items in the shop. To do this, we find the chance that any of the slots contain either the items we want or a miss, then subtract out the chance that all the slots contain a miss. If we're looking for multiple items, we also subtract out the chance that the item contains any lower amount of items to avoid double counting.

$$
\begin{aligned}
P(\text{Only C}) &= (S_\text{C} + S_\text{M})^4 - S_\text{M}^4 \\
&= \left(\frac{1}{2} + \frac{143}{294}\right)^4 - \frac{143}{294}^4 \\
&= \frac{45{,}269{,}717}{50{,}824{,}368} \\
P(\text{Only J}) &= (S_\text{J} + S_\text{M})^4 - S_\text{M}^4 \\
&= \left(\frac{1}{147} + \frac{143}{294}\right)^4 - \frac{143}{294}^4 \\
&= \frac{165{,}896}{51{,}883{,}209} \\
P(\text{Only F}) &= (S_\text{F} + S_\text{M})^4 - S_\text{M}^4 \\
&= \frac{165{,}896}{51{,}883{,}209} \\
P(\text{C and J}) &= (S_\text{C} + S_\text{J} + S_\text{M})^4 - P(\text{Only C}) - P(\text{Only J}) - S_\text{M}^4 \\
&= \left(\frac{1}{2} + \frac{1}{147} + \frac{143}{294}\right)^4 - \frac{45{,}269{,}717}{50{,}824{,}368} - \frac{165{,}896}{51{,}883{,}209} - \frac{143}{294}^4 \\
&= \frac{10{,}523}{453{,}789} \\
P(\text{C and F}) &= (S_\text{C} + S_\text{F} + S_\text{M})^4 - P(\text{Only C}) - P(\text{Only F}) - S_\text{M}^4 \\
&= \frac{10{,}523}{453{,}789} \\
P(\text{J and F}) &= (S_\text{J} + S_\text{F} + S_\text{M})^4 - P(\text{Only J}) - P(\text{Only F}) - S_\text{M}^4 \\
&= \left(\frac{2}{147} + \frac{143}{294}\right)^4 - \frac{2 \cdot 165{,}896}{51{,}883{,}209} - \frac{143}{294}^4 \\
&= \frac{9{,}011}{66{,}706{,}983} \\
P(\text{All}) &= 1 - P(\text{C and J}) - P(\text{C and F}) - P(\text{J and F}) - P(\text{Only C}) - P(\text{Only J}) - P(\text{Only F}) - S_\text{M}^4 \\
&= 1 - \frac{2 \cdot 10{,}523}{453{,}789} - \frac{9{,}011}{66{,}706{,}983} - \frac{45{,}269{,}717}{50{,}824{,}368} - \frac{2 \cdot 165{,}896}{51{,}883{,}209} - \frac{143}{294}^4 \\
&= \frac{437}{1{,}058{,}841}
\end{aligned}
$$

For brevity, we'll modify these equations in the following way. Each subscipt of an F under P (i.e. $P_F(\text{Only J})$) will denote a failure of the Judgement to produce a Chaos the Clown. Each subscript of a S will denote a success.

$$
\begin{aligned}
P_F(\text{Any}) &= 0.3 \cdot P(\text{Any}) \\
P_S(\text{Any}) &= 0.7 \cdot P(\text{Any}) \\
P_{FF}(\text{Any}) &= 0.3^2 \cdot P(\text{Any}) &&= 0.09 \cdot P(\text{Any}) \\
P_{SS}(\text{Any}) &= 0.7^2 \cdot P(\text{Any}) &&= 0.47 \cdot P(\text{Any}) \\
P_{SF}(\text{Any}) &= 0.7 \cdot 0.3 \cdot P(\text{Any}) &&= 0.21 \cdot P(\text{Any})
\end{aligned}
$$

3.1.4 Traffic from the Default State

Now, we find the traffic from every state into every other state. Not every state is reachable from every other state. The Default state is always reachable, but F_2 can only be reached from itself and the default state. We'll begin by finding out how the default state behaves.

$$
\begin{aligned}
D \Rightarrow D &= S_{\mathrm{M}}^4 + (P_F(\text{Only J}) + P_F(\text{Only F})) + P_{FF}(\text{J and F}) \\
&= \frac{418{,}161{,}601}{7{,}471{,}182{,}096} + \frac{2 \cdot 0.3 \cdot 165{,}896}{51{,}883{,}209} + \frac{0.09 \cdot 9{,}011}{66{,}706{,}983} \\
&= \frac{1{,}544{,}949{,}451}{26{,}682{,}793{,}200} \\
D \Rightarrow F_0 &= P(\text{Only C}) + P_S(\text{Only J}) + P_S(\text{Only F}) + P_{SF}(\text{J and F}) \\
&= \frac{45{,}269{,}717}{50{,}824{,}368} + \frac{2 \cdot 0.7 \cdot 165{,}896}{51{,}883{,}209} + \frac{0.21 \cdot 9{,}011}{66{,}706{,}983} \\
&= \frac{7{,}962{,}267{,}823}{8{,}894{,}264{,}400} \\
D \Rightarrow F_1 &= P(\text{C and J}) + P(\text{C and F}) + P_S(\text{J and F}) \\
&= \frac{2 \cdot 10{,}523}{453{,}789} + \frac{0.7 \cdot 9{,}011}{66{,}706{,}983} \\
&= \frac{4{,}428{,}671}{95{,}295{,}690} \\
D \Rightarrow F_2 &= P(\text{All}) \\
&= \frac{437}{1{,}058{,}841}
\end{aligned}
$$

3.1.5 Traffic from the Free States

When we move on to other states, the values shift. Chaos the Clown can no longer spawn in the shop. This subtly changes the values of something like $P(\text{Only J})$, as it no longer needs to account for the case where it hits a Chaos the Clown **and** a Judgement. We'll recalculate the affected probabilities.

$$
\begin{aligned}
P(\text{Only J})_c &= (S_{\mathrm{J}} + S_{\mathrm{Mc}})^4 - S_{\mathrm{Mc}}^4 \\
&= \left(\frac{1}{147} + \frac{145}{147}\right)^4 - \frac{145}{147}^4 \\
&= \frac{4{,}107{,}077}{155{,}649{,}627} \\
P(\text{Only F})_c &= (S_{\mathrm{F}} + S_{\mathrm{Mc}})^4 - S_{\mathrm{Mc}}^4 \\
&= \frac{4{,}107{,}077}{155{,}649{,}627} \\
P(\text{J and F})_c &= 1 - P(\text{Only J})_c - P(\text{Only F})_c - S_{\mathrm{Mc}}^4 \\
&= 1 - \frac{2 \cdot 4{,}107{,}077}{155{,}649{,}627} - \frac{145}{147}^4 \\
&= \frac{36{,}542}{66{,}706{,}983}
\end{aligned}
$$

Now we can find the exiting traffic of the Free state.

$$
\begin{aligned}
F_0 \Rightarrow D &= S_{\mathrm{Mc}}^4 + (P_F(\text{Only J})_c + P_F(\text{Only F}))_c + P_{FF}(\text{J and F})_c \\
&= \frac{442{,}050{,}625}{466{,}948{,}881} + \frac{0.3 \cdot 2 \cdot 4{,}107{,}077}{155{,}649{,}627} + \frac{0.09 \cdot 36{,}542}{66{,}706{,}983} \\
&= \frac{3{,}210{,}474{,}179}{3{,}335{,}349{,}150} \\
F_0 \Rightarrow F_0 &= P_S(\text{Only J})_c + P_S(\text{Only F})_c + P_{SF}(\text{J and F})_c \\
&= \frac{0.7 \cdot 2 \cdot 4{,}107{,}077}{155{,}649{,}627} + \frac{0.21 \cdot 36{,}542}{66{,}706{,}983} \\
&= \frac{13{,}732{,}889}{370{,}594{,}350} \\
F_0 \Rightarrow F_1 &= P_S(\text{J and F})_c \\
&= \frac{0.7 \cdot 36{,}542}{66{,}706{,}983} \\
&= \frac{18{,}271}{47{,}647{,}845}
\end{aligned}
$$

Once again, the values shift. Judgements can no longer spawn in the shop, and we'll isolate total shop misses in order to apply our previous convention of P_F and P_S to misses coming from our inventory.

$$
\begin{aligned}
P(\text{F})_j &= 1 - S_{\mathrm{Mj}}^4 \\
&= 1 - \frac{139}{140}^4 \\
&= \frac{10{,}858{,}959}{384{,}160{,}000} \\
P(\text{Miss})_j &= S_{\mathrm{Mj}}^4 \\
&= \frac{373{,}301{,}041}{384{,}160{,}000}
\end{aligned}
$$

Both states with items in the inventory behave similarly, so we can do them both at once.

$$
\begin{aligned}
F_1 \Rightarrow D &= P_F(\text{Miss})_j + P_{FF}(\text{F}))_j \\
&= \frac{0.3 \cdot 373{,}301{,}041}{384{,}160{,}000} + \frac{0.09 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{1{,}613{,}823{,}123}{5{,}488{,}000{,}000} \\
F_1 \Rightarrow F_0 &= P_S(\text{Miss})_j + P_{SF}(\text{F}))_j \\
&= \frac{0.7 \cdot 373{,}301{,}041}{384{,}160{,}000} + \frac{0.21 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{537{,}941{,}041}{784{,}000{,}000} \\
F_1 \Rightarrow F_1 &= P_S(\text{F})_j \\
&= \frac{0.7 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{10{,}858{,}959}{548{,}800{,}000} \\
F_2 \Rightarrow D &= P_{FF}(\text{Miss})_j + P_{FFF}(\text{F}))_j \\
&= \frac{0.09 \cdot 373{,}301{,}041}{384{,}160{,}000} + \frac{0.027 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{4{,}841{,}469{,}369}{54{,}880{,}000{,}000} \\
F_2 \Rightarrow F_0 &= P_{SF}(\text{Miss})_j + P_{FFS}(\text{F}))_j \\
&= \frac{0.21 \cdot 373{,}301{,}041}{384{,}160{,}000} + \frac{0.063 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{1{,}613{,}823{,}123}{7{,}840{,}000{,}000} \\
F_2 \Rightarrow F_1 &= P_S(\text{Miss})_j + P_{FS}(\text{F}))_j \\
&= \frac{0.7 \cdot 373{,}301{,}041}{384{,}160{,}000} + \frac{0.21 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{537{,}941{,}041}{784{,}000{,}000} \\
F_2 \Rightarrow F_2 &= P_S(\text{F}))_j \\
&= \frac{0.7 \cdot 10{,}858{,}959}{384{,}160{,}000} \\
&= \frac{10{,}858{,}959}{548{,}800{,}000}
\end{aligned}
$$

3.1.6 Incoming Traffic for Every State

We now have the full probabilities of every state entering every other state. If we lay them all out, we can find the incoming traffic of every state.

$$
\text{Total Traffic} = \begin{cases}
D &= (D \Rightarrow D) \cdot D + (F_0 \Rightarrow D) \cdot F_0 + (F_1 \Rightarrow D) \cdot F_1 + (F_2 \Rightarrow D) \cdot F_2 \\
F_0 &= (D \Rightarrow F_0) \cdot D + (F_0 \Rightarrow F_0) \cdot F_0 + (F_1 \Rightarrow F_0) \cdot F_1 + (F_2 \Rightarrow F_0) \cdot F_2 \\
F_1 &= (D \Rightarrow F_1) \cdot D + (F_0 \Rightarrow F_1) \cdot F_0 + (F_1 \Rightarrow F_1) \cdot F_1 + (F_2 \Rightarrow F_1) \cdot F_2 \\
F_2 &= (D \Rightarrow F_2) \cdot D + (F_2 \Rightarrow F_2) \cdot F_2
\end{cases}
$$

Here, the values D, F_0, F_1, and F_2 represent the total traffic entering these states. To quickly run through an example, the total traffic to the F_2 state would be the probability that it is entered through the default state multiplied by the total traffic to the default state, and the chance that it enters itself multiplied by its own total traffic.

To better understand this, let's look at at hypothetical. There are only two states, A and B. A enters B 50% of the time, and itself %50 of the time. B always enters A. $A = (A \Rightarrow A) \cdot A + (B \Rightarrow$

$A) \cdot B$. We set A to 1 to see how this behaves. The equation becomes $1 = 0.5 \cdot 1 + 1 \cdot B$, so B is 0.5. The total traffic of this state is $A + B$, which is 1.5, which makes sense intuitively.

We've found the expected probabilities that show how likely a state is to enter any other state (i.e. $D \Rightarrow F_2$), now we need to find D, F_0, F_1, and F_2.

3.1.7 Substituting in our Found Values

We are only concerned with what happens for a single paid reroll. Our goal here is to find out how many shops, or shop states, we'll be expected to see for that paid reroll. To do this, we can set the traffic of D to 1, and then use that as a lever to find the expected traffic to every other state.

$$
\begin{aligned}
D &= 1 \\
F_2 &= (D \Rightarrow F_2) \cdot D + (F_2 \Rightarrow F_2) \cdot F_2 \\
&= \frac{437}{1{,}058{,}841} \cdot 1 + \frac{10{,}858{,}959}{548{,}800{,}000} \cdot F_2 \\
&= \frac{699{,}200{,}000}{1{,}660{,}623{,}993{,}567} \\
F_0 &= (D \Rightarrow F_0) \cdot D + (F_0 \Rightarrow F_0) \cdot F_0 + (F_1 \Rightarrow F_0) \cdot F_1 + (F_2 \Rightarrow F_0) \cdot F_2 \\
&= \frac{7{,}962{,}267{,}823}{8{,}894{,}264{,}400} \cdot 1 + \frac{13{,}732{,}889}{370{,}594{,}350} \cdot F_0 + \frac{537{,}941{,}041}{784{,}000{,}000} \cdot F_1 + \frac{1{,}613{,}823{,}123}{7{,}840{,}000{,}000} \cdot \frac{699{,}200{,}000}{1{,}660{,}623{,}993{,}567} \\
&= \frac{7{,}397{,}325{,}062{,}253}{10{,}381{,}424{,}320{,}000} \cdot F_1 + \frac{7{,}963{,}038{,}691}{8{,}564{,}675{,}064} \\
F_1 &= (D \Rightarrow F_1) \cdot D + (F_0 \Rightarrow F_1) \cdot F_0 + (F_1 \Rightarrow F_1) \cdot F_1 + (F_2 \Rightarrow F_1) \cdot F_2 \\
&= \frac{4{,}428{,}671}{95{,}295{,}690} \cdot 1 + \frac{18{,}271}{47{,}647{,}845} \cdot F_0 + \frac{10{,}858{,}959}{548{,}800{,}000} \cdot F_1 + \frac{537{,}941{,}041}{784{,}000{,}000} \cdot \frac{699{,}200{,}000}{1{,}660{,}623{,}993{,}567} \\
&= \frac{998{,}880{,}714{,}529{,}700{,}000}{20{,}774{,}133{,}579{,}956{,}225{,}931} \\
F_0 &= \frac{4{,}952{,}506{,}333}{5{,}137{,}372{,}592}
\end{aligned}
$$

3.1.8 The Final Expected Rerolls from every Paid Reroll

Now that we have the expected traffic of every state from a single paid reroll, we add them all together and find out how many shop states we'll see from a single paid reroll.

$$
\begin{aligned}
\text{Total Seen Shop States} &= D + F_0 + F_1 + F_2 \\
&= 1 + \frac{4{,}952{,}506{,}333}{5{,}137{,}372{,}592} + \frac{998{,}880{,}714{,}529{,}700{,}000}{20{,}774{,}133{,}579{,}956{,}225{,}931} + \frac{699{,}200{,}000}{1{,}660{,}623{,}993{,}567} \\
&= \frac{4{,}682{,}534{,}754{,}379{,}326{,}929{,}675}{2{,}326{,}702{,}960{,}955{,}097{,}304{,}272} \\
&\approx 2.01
\end{aligned}
$$

3.1.9 Judgement Failures: Ancillary benefits

Failures of Judgements to produce a Chaos still provide you with benefits. When a Judgement rolls, it has a $\frac{1}{4}$ chance to produce an Uncommon. We'd like to find out how many Double Tags we're expected to receive from these failures.

First, we find out how many Judgements we'll be using. Our expected rerolls without using Judgements is actually incredibly easy to find. We just take a look at a single paid reroll, and see if it rolls a Common joker. If it does we get a free reroll which will never have a Chaos the Clown, so it will always return to the paid state right afterwards. This means our expected rolls is simply the sum of 1 and the chance of finding a Chaos in a given shop, or $(1 - S_C)^4$.

$$\begin{aligned}\text{Total Seen Shop States}_{\text{No Judgement}} &= 1 + \text{Chance of Chaos} \\ &= 1 + \left(\frac{1}{2}\right)^4 = \frac{31}{16}\end{aligned}$$

If we find the difference between our rerolls using Judgements, and our rerolls without, the result is the expected amount of Judgements per paid reroll multiplied by the chance of a joker being Common. This gets converted to the expected amount of Judgements per any reroll.

$$\begin{aligned}\text{Expected Judgements producing Commons} &= \text{Total Seen Shop States} - \text{Total Seen Shop States}_{\text{No Judgement}} \\ &= \frac{4{,}682{,}534{,}754{,}379{,}326{,}929{,}675}{2{,}326{,}702{,}960{,}955{,}097{,}304{,}272} - \frac{31}{32} \\ &= \frac{21{,}818{,}470{,}941{,}103{,}237{,}831}{290{,}837{,}870{,}119{,}387{,}163{,}034} \\ \text{Expected Judgements Per Paid} &= \text{Expected Judgements Producing Commons} \cdot \frac{10}{7} \\ &= \frac{109{,}092{,}354{,}705{,}516{,}189{,}155}{1{,}017{,}932{,}545{,}417{,}855{,}070{,}619} \\ &\approx 0.11\end{aligned}$$

3.2 Diet Colas from Rolling

We now have our total Shops seen and our total Judgements used. We need to convert this into total Double Tags received per paid shop.

First, let's set some boundary conditions. We'll need to grab every Uncommon joker we can, which is all of them minus Diet Cola and Showman. Madness, Seltzer, and Turtle Bean provide challenges here, but as we're only considering negative jokers as useful we can assume that either the player picks up these in negative form, or they use their non-negative slots to hold copies of them. This means that every Uncommon we see will have a straight $\frac{1}{2}$ chance of being a Cola, and if we see two we will always see a Cola.

The binomial formula comes to the rescue here.

$$\begin{aligned}P(\text{Slot is Uncommon}) &= \frac{20}{28} \cdot \frac{1}{4} = \frac{5}{28} \\ P(\text{Cola in Shop}) &= \frac{1}{2} \cdot P(\text{1 Uncommon}) + P(\text{2-4 Uncommons}) \\ &= \frac{1}{2} \cdot \binom{4}{1} \cdot \left(\frac{5}{28}\right)^1 \cdot \left(1 - \frac{5}{28}\right)^{4-1} + \sum_{i=2}^{4} \left[\binom{4}{i} \cdot \left(\frac{5}{28}\right)^i \cdot \left(1 - \frac{5}{28}\right)^{4-i}\right] \\ &= \frac{213{,}145}{614{,}656} \\ P(\text{Cola in Paid Reroll})_{\text{Shop}} &= \text{Total Seen Shops from Paid Reroll} \cdot P(\text{Cola in Shop}) \\ &= \frac{4{,}682{,}534{,}754{,}379{,}326{,}929{,}675}{2{,}326{,}702{,}960{,}955{,}097{,}304{,}272} \cdot \frac{213{,}145}{614{,}656} \\ &= \frac{998{,}058{,}870{,}222{,}181{,}638{,}425{,}577{,}875}{1{,}430{,}121{,}935{,}168{,}816{,}288{,}654{,}610{,}432}\end{aligned}$$

Now, we'll find the chance that any Judgement used will produce a Diet Cola. This time, we need to see if a Showman has spawned in the shop, as if it has and the Judgement has rolled an Uncommon, it is guaranteed to be a Diet Cola. The chance of a Showman in the shop happens to be the same as the chance of a Cola in the shop.

$$
\begin{aligned}
P(\text{Judgement is Cola}) &= P(\text{Uncommon}) \cdot \left(\frac{1}{2} \cdot P(\text{No Showman in Shop}) + P(\text{Showman in Shop}) \right) \\
&= \frac{1}{4} \cdot \left(\frac{1}{2} \cdot \left(1 - \frac{213{,}145}{614{,}656} \right) + \frac{213{,}145}{614{,}656} \right) \\
&= \frac{827{,}801}{4{,}917{,}248} \\
P(\text{Cola in Paid Reroll})_{\text{Judgement}} &= \text{Expected Judgements per Paid} \cdot P(\text{Judgement is Cola}) \\
&= \frac{109{,}092{,}354{,}705{,}516{,}189{,}155}{1{,}017{,}932{,}545{,}417{,}855{,}070{,}619} \cdot \frac{827{,}801}{4{,}917{,}248} \\
&= \frac{90{,}306{,}760{,}317{,}581{,}006{,}898{,}698{,}155}{5{,}005{,}426{,}773{,}090{,}857{,}010{,}291{,}136{,}512}
\end{aligned}
$$

Now we can see exactly how many Double Tags we're expected to see per paid reroll. Simply add the amount we'll get from the shop to the amount we'll get from Judgements.

$$
\begin{aligned}
\text{Total Double Tags from Reroll}(\tau) &= P(\text{Cola in Paid Reroll})_{\text{Shop}} + P(\text{Cola in Paid Reroll})_{\text{Judgement}} \\
&= \frac{998{,}058{,}870{,}222{,}181{,}638{,}425{,}577{,}875}{1{,}430{,}121{,}935{,}168{,}816{,}288{,}654{,}610{,}432} + \frac{90{,}306{,}760{,}317{,}581{,}006{,}898{,}698{,}155}{5{,}005{,}426{,}773{,}090{,}857{,}010{,}291{,}136{,}512} \\
&= \frac{7{,}167{,}025{,}612{,}190{,}433{,}482{,}776{,}441{,}435}{10{,}010{,}853{,}546{,}181{,}714{,}020{,}582{,}273{,}024} \\
&\approx 0.71
\end{aligned}
$$

3.3 Money from Negative Skips

A fundamental problem with this setup is that it never grows until you skip a round. You simply accumulate double tags, cashing them in can only happen at specific times. Let's look at the ideal case.

We'll assume every Ante has a negative skip in the Small Blind. Let's look at how that would go.

1. Skip the Small Blind
2. Play the Big Blind
   (a) Money generated is equal to the amount made last round.(Money_{r-1})
3. Enter the Big Blind Shop.
4. Receive the Negative Jokers. Roll for new tags using Money_{r-1}
5. Play the Boss Blind
   (a) Money generated is equal to the amount made this round.(Money_{r})
6. Enter the Boss Blind Shop.
7. Roll for new tags using Money_{r}
8. Repeat

This means that every three rounds, the amount of negative tags we'll receive is the amount of Double Tags obtained from Money_{r-1} and Money_{r}. If the skip happens further than three rounds out, then you repeat steps 5-7 until the negative skip is present.

3.3.1 Revisiting our Money Generation

We know from our previous work the amount of money we can make from any pool of negative jokers, and how to get out our current pool of negative jokers from the amount of money we're generating.

$$\text{Money} = \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \text{J}^4$$

$$\text{J} = \sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}}$$

Now, we need to add in the new jokers. We're rolling i times, where i is the interval between negative skips. For the example we gave, i would be 3.

$$\begin{aligned}
\text{Slots}_r &= 2\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right) \\
\text{Rerolls}_r &= \frac{\sqrt{8 \cdot \text{Money}_r + 1} + 1}{2} \\
\text{J}_{\text{new}} &= \text{Total Double Tags from Reroll}(\tau) \cdot ((i-2) \cdot \text{Rerolls}_r + \text{Rerolls}_{r-i}) \\
&= \tau \cdot \left(\frac{(i-2) \cdot \sqrt{8 \cdot \text{Money}_r + 1} + \sqrt{8 \cdot \text{Money}_{r-i} + 1} + (i-1)}{2}\right)
\end{aligned}$$

$$\boxed{\text{Money}_{r+i} = \frac{1{,}001}{1{,}620{,}000{,}000} \cdot \left(\sqrt[4]{\frac{1{,}620{,}000{,}000}{1{,}001} \cdot \text{Money}_r} + \tau \cdot \left(\frac{(i-2) \cdot \sqrt{8 \cdot \text{Money}_r + 1} + \sqrt{8 \cdot \text{Money}_{r-i} + 1} + (i-1)}{2}\right)\right)^4}$$

I was unable to work out a way to solve this in the generalized terms of Money_0 and Money_r. We can tell a significant amount from this recurrence relation even without a solved growth function.

The major difference between this relation and our old one is that this new relation looks significantly better. They're both extremely similar, as our method of gaining money hasn't changed, but in the old equation we were multiplying expected Shop Slots by $\frac{3}{1400}$ while in this new one we are multiplying our rerolls by τ. Once you account for the differences between slots and rerolls, our new multiplier is ~ 83x better at generating negative jokers.

This sounds strictly superior, but there is a problem.

3.4 Tipping Point

Our new equation only scales every 3 rounds at best. For some initial value Money_0 the old equation will surpass the new one over three rounds. This is the tipping point.

This is where we finally depart from followable and exact mathematics, and just simulate the answer. The constraints of needing both Money_r and Money_{r-i} to find Money_{r+i} has proven too much for me to figure out. Simulation and imprecise numbers will have to do.

The most ideal case for this formula occurs when negative skips happen once per Ante, or when $i = 3$. Skips cannot happen earlier, as you need to see at least a single shop between skips to accrue double tags again and both possible skips in an Ante happen back-to-back. We will discuss what happens for non-ideal cases later.

To simulate this, we begin with a seed value at a very low number, say \$10, and say that Money_r and Money_{r-i} are both equal for the very first cycle. This tiny imperfection will correct itself as we go through the cycles. Once the growth formula utilizing a Showman outpaces the Cola formula over the three rounds it takes to scale, we stop, log the value of Money_0 where that occurred, and then go again with an incremented seed value. We save the lowest found value, telling us when we should stop using Diet Colas and begin rolling with a Showman.

Once changing the precision and range of the increment failed to change the resultant tipping point, we end up with our answer.

$$\boxed{f_{showman}(\text{Money}_{r+i}) > f_{cola}(\text{Money}_{r+i}) \text{ at } \text{Money}_0 \approx \$1.96 \times 10^{14} \text{ for all circumstances}}$$

This assumes perfectly placed negative skips. Extending the window of negative skips by even a single round, so that they come every fourth round instead of every third, drops this value to 1.58×10^{13}. If we assume we have a negative skip every other Ante instead of every Ante, it drops to 7.8×10^{11}, just below the trillions. Every third Ante results in a value of 6.33×10^{10}, under $100 billion.

If we were to assume you find negative skips at the rate you should expect to get them, which is once every 12 Antes, this value plummets to $67,195,123. This is the tipping point for average luck. In real circumstances you aren't guaranteed to see a single negative skip across the 39-41 total reachable Antes and this growth function will never grow.

4 Rare Tag Duplication Glitch: Impractical Tech

A glitch that isn't very well known is Rare Tag Duplication. It's a powerful glitch with the only true caveat being the time it forces players to spend in shops to get full value out of it. The requirements to set it up are low, and the results are staggering.

First, get more Rare Tags than your opening shop has shop slots, preferably at least double the amount. Reroll the shop once, then exit to the main menu and reenter. You'll notice that the tags consumed after you rerolled are refunded, and the rare jokers that the tags spawned are now at full price. You can now reroll again, and repeat. The only jokers that you'll see in this entire shop will be rares, all at the cost of any Rare Tags that were spent during your very first shop slots.

This requires that you exit to the menu and reenter every single time you roll, which makes it very impractical when you roll absurd amounts of times. However, it does completely blow open our money generation formula.

4.1 Readjusting for when the Shop can only Spawn Rares

We're going to use the money generation formula that assumes we don't have an infinite deck here, as Rare jokers don't increase your hand size at all outside of card generation. We've already done this adjustment several times, so I'll just run through this quickly.

$$\text{Rare}_{\text{Rare Tag}} = \text{Slots} \cdot \frac{20}{28} \cdot \frac{3}{1000} \cdot 1 \cdot \frac{1}{20} = \frac{3}{28{,}000}\text{Slots}$$

$$\text{Money} = \frac{1}{2} \cdot \underbrace{(\text{DNA} + 2 \cdot \text{Copy})^2}_{\text{Copying Certificate(s) and DNAs}} \cdot \underbrace{\left(\frac{1}{2} \cdot \text{Copy}\right)}_{\text{Reserved}} \cdot \underbrace{\left(\frac{1}{2} \cdot \text{Copy}\right)}_{\text{Retriggers}}$$

$$= \frac{1}{2} \cdot \left(\frac{3}{5{,}600} \cdot \text{Slots}\right)^2 \cdot \left(\frac{3}{56{,}000} \cdot \text{Slots}\right)^2$$

$$= \frac{81}{196{,}689{,}920{,}000{,}000{,}000} \cdot \text{Slots}^4$$

$$= \frac{1}{51{,}200} \cdot \text{J}^4$$

$$\text{Money}_{r+1} = \frac{1}{51{,}200} \cdot \left(\sqrt[4]{51{,}200 \cdot \text{Money}_r} + \frac{3}{700}\left(\sqrt{8 \cdot \text{Money}_r + 1} + 1\right)\right)^4$$

$$X = \frac{1}{51{,}200} \cdot \frac{3^4 \cdot \left(\sqrt{8}^4\right)}{700^4} \approx 4.22 \times 10^{-13}$$

$$\boxed{\text{Money}_r \approx (2.37 \times 10^{12}) \cdot \left(\frac{\text{Money}_0}{2.37 \times 10^{12}}\right)^{2^r}}$$

This begins scaling at values $\sim$ 31x lower than our theoretical money scaling, a significant improvement.

5 Problems in Creating a Run Where This is Useful

Balatro players don't think of money as growing in this fashion. Almost all Balatro players find it incredibly odd to make money over $1,000 a round, never mind $1 billion. Let's go over everything that stops this equation from impacting a regular run.

5.1 Initial Startup Costs

These equations don't start meaningfully growing until you're at around $100 billion a round. Reaching that is a gargantuan task. There are two known methods of getting to this value.

5.1.1 Perkeo

Perkeo has an infinite. Get a negative version of every tarot besides the Emperor, Fool, and one of the following [Temperance|Hermit|Judgement] and you have infinite money, so long as you get the loop going. You can use this to get your run to a state where our equation for money generation starts coming into play.

5.1.2 Bean Bug

The furthest point the Bean Bug has been taken in a run is reaching a hand size of 209 post-Ante 8. After using 13 ectoplasms, this means you can make $585,000 a round, earning >9 negatives a round. Use these negatives to set up for our Double Tag growth formula. Once you successfully find two skips with early timings, you'll be earning somewhere in the realm of $1 trillion a round, which after 5 Antes, would theoretically give you naneinf money.

5.2 Performance Issues

Balatro as a game is not made for this. You'll experience lag that fully halts the game, alongside several crashes from various sources, before being able to make this amount of money. Let's look at everything that will stop you, and potential fixes, as we progress.

5.2.1 Reserved Parking is a Laggy Money Generator

Reserved Parking gives you money in the smallest possible increment, +1 at a time, and runs an animation and a random number generator (even if you have an Oops All Sixes) every time it decides to pay you out. Very quickly, your game will grind to a halt.

This can be solved (for now) with a mod like Handy, or anything that lets you skip animations.

5.2.2 Rendering of Double Tags

Every single tag you accrue is rendered by the game engine, even the ones offscreen. This will cause enough lag to completely wall you after a while.

This can be solved with a mod like BetterTags, which stacks identical tags graphically.

5.2.3 Memory Storage of Double Tags

BetterTags only changes the graphics of double tags. They are still stored as individual objects by the game, and they need to be to make the saves of the run remain Vanilla. Enough tags will crash the game with a Thread Error, where it will tell you that you've run out of memory.

This can be solved by editing a mod like BetterTags to stack the tags in the games storage. **This will break compatibility with saves that do not use a mod like this.**

5.2.4 The Lag from Reserved Parking Reappears

Simply removing the animations isn't enough. Every trigger of Reserved Parking is still creating a laundry list of calls internally for the game. The lag from this will, once more, grind your game to a halt.

This can be solved by using a mod to completely disable the functionality of Reserved Parking, and then do its own singular calculation that pays out what Reserved Parking would have paid out at the end of all of its calls across all cards and retriggers.

5.2.5 Memory Storage of Jokers

Similarly to the problem with double tags, jokers are stored as individual objects. We will be accruing so many of them that this will run into the exact same memory issues that we hit with double tags.

This can be solved with a mod like Yorick, which stacks identical jokers. **This will break compatibility with saves that do not use a mod like this.**

5.3 Time Investment

I've said that this only begins to meaningfully matter at $100 billion dollars. That much money would give you 1,788,854 shop slots. If you rolled once a second, that would take 124.2 hours to get through. Over three weeks of a 9-5 job, just to finish rolling through a single shop.

Most people are not willing to do this.

5.4 Earlier Caps than Naneinf

Score caps at e308, as we all know. This is the maximum value that a 64 bit float can store. This is only achievable because score has multipliers. Money, on the other hand, can only be earned in discrete amounts. Reserved parking can be amplified multiplicatively but every time it interacts with our money pool, it adds a single dollar at a time.

This runs into a new limitation of floats, precision loss. Floats use 2^{64} bits to store values with an upper limit of 2^{1024}. In order to do that, floats don't store these values perfectly. At certain limits

it rounds them to the closest number it can store, which sometimes loses information.

At 9.01e15, adding +1 to our money no longer changes its value. The addition gets rounded away. If we were to use Golden Ticket, which adds money in +4 intervals, this limit gets bumped to 3.6e16. If we were to use Temperances, which add money in +50 intervals, it gets bumped to 1.15e18. The highest known limit that can be reached is with Cloud 9, which will get capped at the same limit our money from Reserved Parking gets capped at, 9.01e15. This means every time it interacts with our money pool, it directly adds 9.01e15 to it. If you had an incredible number of rounds, where a maxed-out Cloud 9 could trigger near-endlessly, it would get capped at 8.11e31.